

Ej	1	2	3
	B	B	B

Aprobado

1 de 4
Corrigió
DanielaEjercicio 1

a)

Camino (bin(nil, R, nil), [R]).

Camino (bin(I, R, ~~D~~), [R|C]) :- ~~Existe un camino~~ camino(I, C).Camino (bin(~~I~~, R, D), [R|C]) :- camino(D, C).

↳ falta pedir que los subárboles no sean nil, si no genera

b) resultados repetidos y soluciones no deseadas

Camino Mas Largo (A, C) :- camino(A, C), length(C, L),
not (masLargoHayUnoMasLargo(A, L)).

masLargoHayUnoMasLargo(A, L) :- camino(A, C), length(C, L2), L2 > L.

c)

CaminoDeLong(A, N) :- camino(A, C), length(C, N).

~~CaminoUnicoDeLong(A, N) :- camino(A, C), length(C, N),~~~~caminoUnicoDeLong(A, N) :- camino(A, C), length(C, N),~~

caminoUnicoDeLong(bin(nil, R, nil), 1, [R]).

CaminoUnicoDeLong(bin(I, R, D), N, [R|C]) :- ~~N > 1~~ N > 1, N1 is N-1,
caminoUnicoDeLong(I, N1, C),
not (caminoDeLong(D, N1)).CaminoUnicoDeLong(bin(I, R, D), N, [R|C]) :- N > 1, N1 is N-1,
caminoUnicoDeLong(D, N1, C),
not (caminoDeLong(I, N1)).No hacía falta separar por
casos, podías usar un solo not

Ejercicio 2

a) Notación: $X \rightarrow Y$ significa X es prototipo de Y directo

La relación de prototipos es transitiva $X \rightarrow Y \rightarrow Z$ entonces X es prototipo de Z .

Slime. Prototype \rightarrow Slime.Sonador.prototype

\downarrow
b
 \downarrow
c

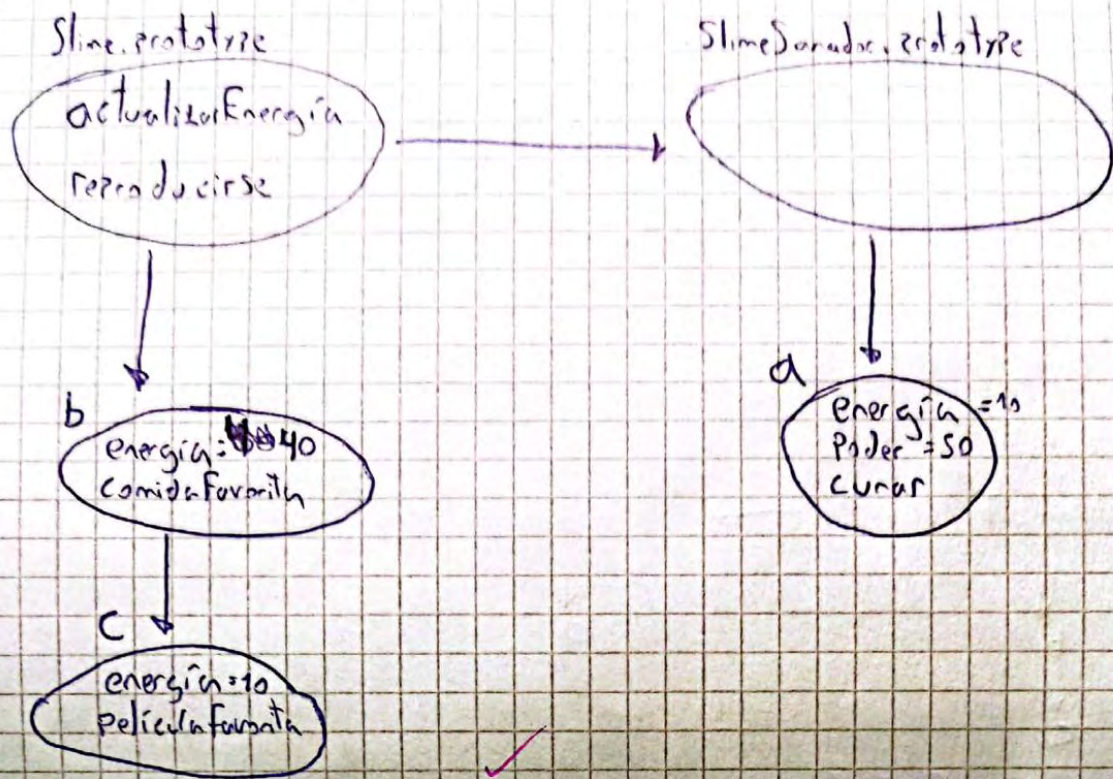
\downarrow
a

mensaje \ objeto	Slime.prototype	Slime.Sonador.prototype	a	b	c
energía			✓	✓	✓
actualizarEnergía	✓	✓	✓	✓	✓
Poder			✓		
curar			✓		
reproducirse	✓	✓	✓	✓	✓
comida Favorita				✓	✓
Película Favorita					✓

Nota check naranja significa puede responder

check verde significa ~~que el~~ mensaje está definido en ese objeto

Del otro lado en está escrito en modo arbol \rightarrow



Todo objeto puede responder a los mensajes que tiene definidos y a los de sus prototipos.

E. energía devuelve 10 pues primero intenta responder con lo que tiene definido y ~~en caso~~ solo en caso de no tenerlo busca la respuesta en la cadena de ~~del~~ prototipos.

Ejercicio 2

b)

$$\text{longitud} \stackrel{\text{def}}{=} [\text{val} = S(f) \text{ if } f.\text{org} \neq \text{Empty}, \text{ if (cero)} \\ \text{org} = S(f) f.\text{org}] \quad \underbrace{((f.\text{org} = f.\text{org}.\text{tail}).\text{val}.\text{succ})}_{= f(f.\text{org}.\text{tail})}$$

Notar que no puede ser escrito en forma de lambda pues es recursiva (la longitud de la lista vacía es cero, de la no vacía es una más que la de su ~~cola~~ tail).

a)

1. $\{ \text{Frutul}(\text{Frutilla}) \}$
2. $\{ \text{Frutul}(\text{banana}) \}$
3. $\{ \text{cremoso}(\text{banana}) \}$
4. $\{ \text{cremoso}(\text{amer. cona}) \}$
5. $\{ \text{cremoso}(\text{Frutilla}) \}$
6. $\{ \text{leGusta}(x_6), \neg \text{Frutul}(x_6), \neg \text{cremoso}(x_6) \}$
7. $\{ \text{cucurucha}(x_7, y_7), \neg \text{leGusta}(x_7), \neg \text{leGusta}(y_7) \}$
8. $\{ \neg \text{cucurucha}(x_8, y_8) \}$

~~2. Da geht es um die Frage, ob die...~~

6)

8, 7 $\sigma = \{X_7 + X_3, Y_7 + Y_3\}$

9. $\{ \neg \text{leGusta}(x_3), \neg \text{leGusta}(y_3) \}$

9. y 6 $\sigma = \{X_8 + X_6, Y_3 + X_6\}$

10. $\{ \neg \text{Fruta}(X_6), \neg \text{cremoso}(X_6) \}$

10, 2 $\sigma = \{x_6 = \text{banana}\}$

11. $\{ \rightarrow \text{Cremoso (barrena)} \}$

$$11, 3 \quad \sigma = \{ \}$$

Comenzando los MGU:

$X = \text{bananen}$
 $Y = \text{bananen}$

c) No fue SLD ya que al unificar las cláusulas 9 y 6 se utiliza la regla de resolución general en lugar de la binaria (se unifican $\neg \text{leGusta}(X_8)$, $\neg \text{leGusta}(Y_3)$, $\text{leGusta}(X_6)$).

De todas formas el sistema tiene una solución SLD
Pues son todas cláusulas de Horn (a lo sumo un literal positivo),
hay una cláusula objetivo (la 8 tiene todos los literales
negativos), ~~ya el método SLD es completo~~ ya está demostrado
que el conjunto de cláusulas es insatisfacible y el método
SLD es completo.

~~Se puede demostrar que el sistema no tiene solución SLD~~
~~ya que no hay una cláusula objetivo~~
~~el método SLD es completo~~

~~no $\neg \text{leGusta}(X_8)$, $\neg \text{leGusta}(X_6)$, $\text{leGusta}(X_6)$~~

119

Prolog recorre las cláusulas de arriba hacia abajo y unifica siempre
con el ~~primer~~ solo el primer literal de la objetivo. ~~Antes de unificar~~
~~unificamos~~ Prolog hubiera ~~de~~ instanciado X e Y en
fautilla como primera opción.